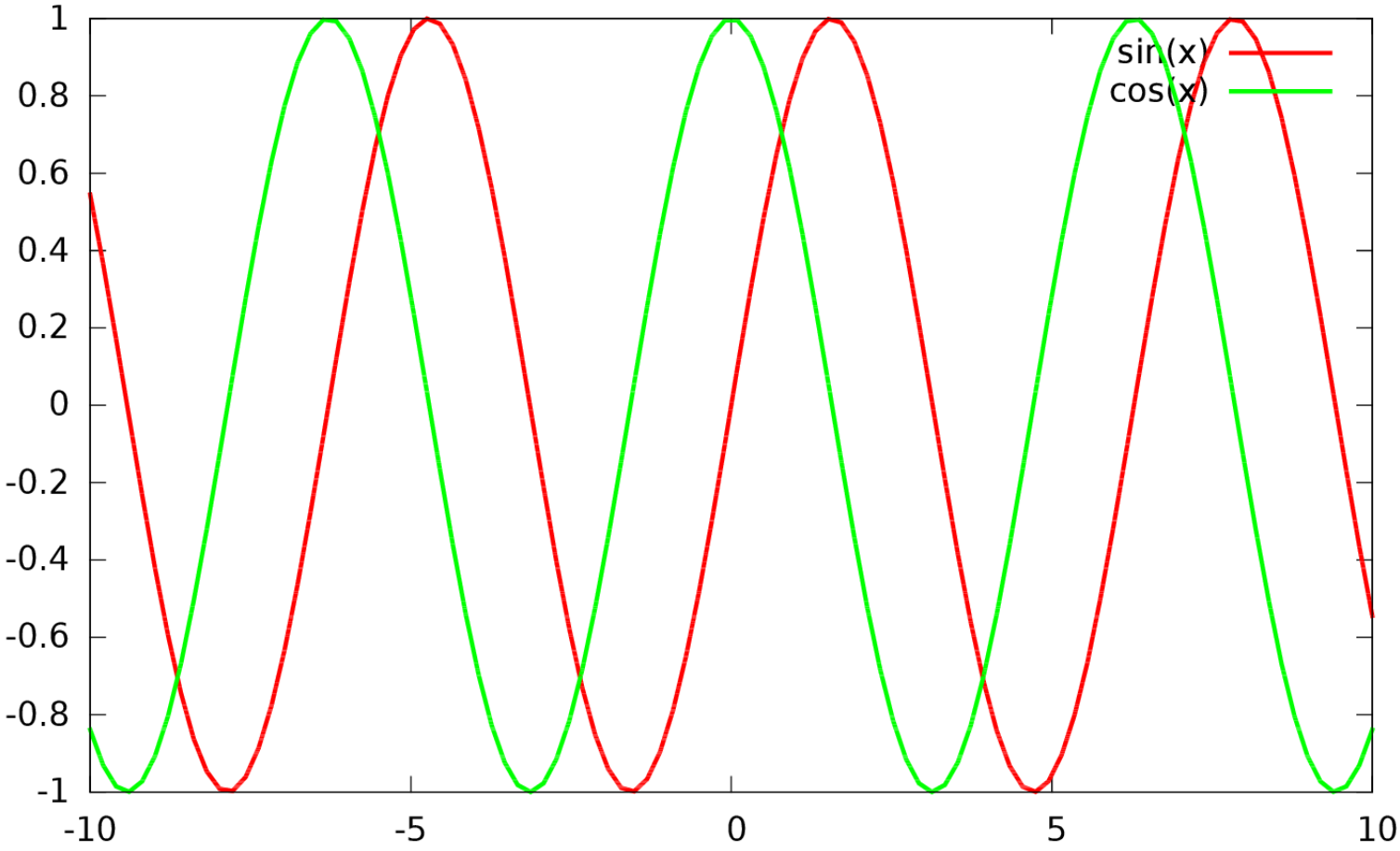
Maintaining accessibility through testing?

Samuel Thibault
Slides & stuff on
http://brl.thefreecat.org/

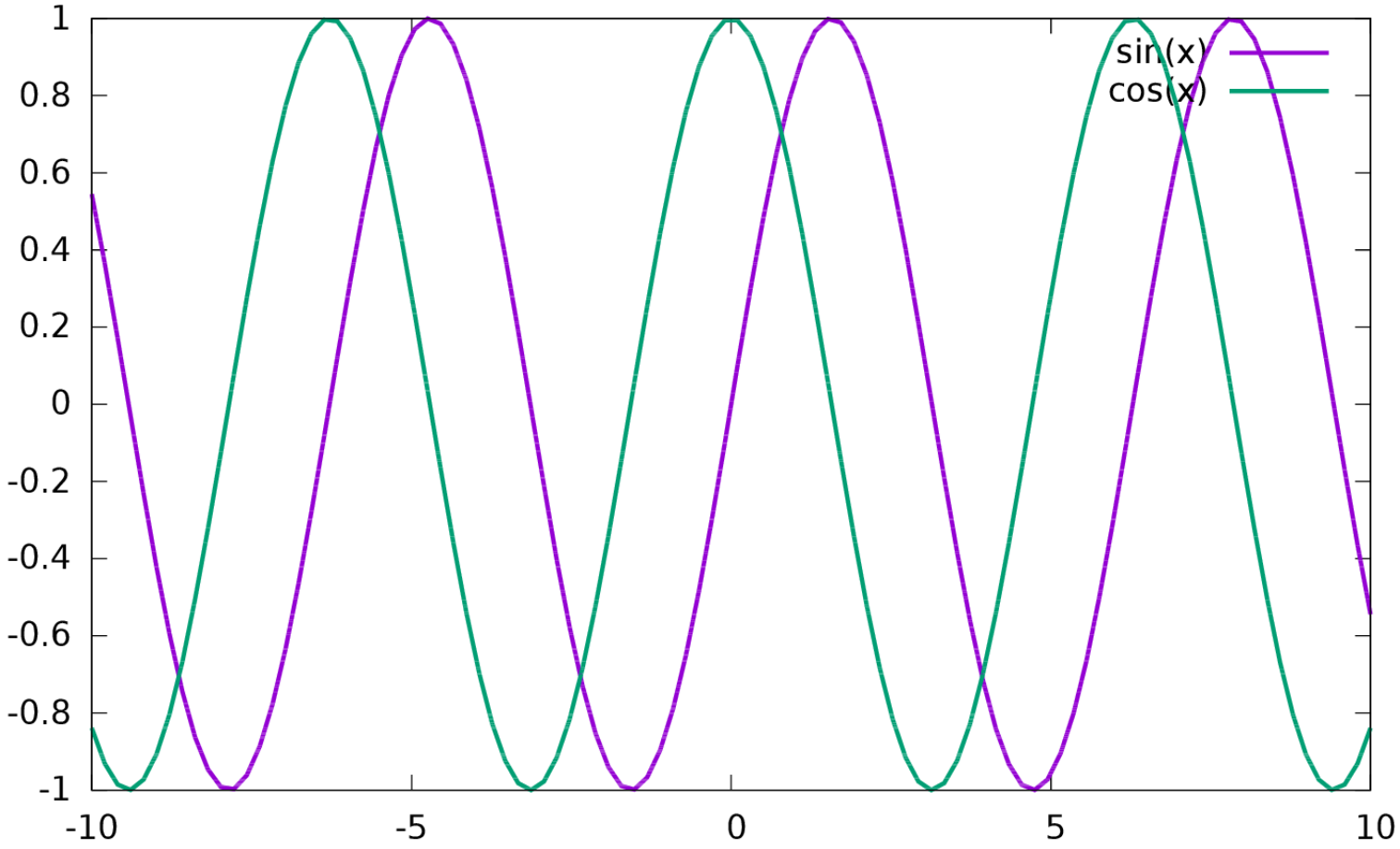
http://hypra.fr/

Gnuplot



Color blindness: 8% male, 0.5% female

Gnuplot 5!!



Color blindness: 8% male, 0.5% female

What is accessibility?

AKA a11y

Usable by everybody, including with specific needs

- Blind, low vision, deaf, …
- Cognition
- Motor
- You

- 10% handicapped, 20% limited (INSEE poll 1254)

This is all about freedom #0

“The freedom to run the program, for any purpose”

- What about being *able to use* the program?

“this is free software, you can modify it” (freedom #1)

- Can. Not. Happen.

A question of priority

- Should be prioritized
  - Just like security and internationalization

A question of who doing it

- Concerns only a small fraction of population
  - Already a hard time using computers...
  - Almost nobody with both disabilities and programming skills (and very difficult to work)
  - Even fewer people with awareness and programming skills

→ “This is free software, you can modify it” can not work.

A question of who doing it

- Paying some people to do it?
  - Means monitoring everything everywhere
    - Not viable, too fast evolutions
  - Would need way too many people
    - With competence in the various software
  - E.g. Joanmarie at Igalia can’t do everything
  - Spending their full workdays debugging accessibility is more than painful
- Support has to be integrated
  - Distributed among maintainers themselves

UNO rights of persons with disabilities

"Discrimination on the basis of disability" means any distinction, exclusion or restriction on the basis of disability which has the purpose or effect of impairing or nullifying the recognition, enjoyment or exercise, on an equal basis with others, of all human rights and fundamental freedoms in the political, economic, social, cultural, civil or any other field. It includes all forms of discrimination**, including denial of reasonable accommodation**

"Reasonable accommodation" means necessary and appropriate modification and adjustments **not imposing a disproportionate or undue burden**, where needed in a particular case, to ensure to persons with disabilities the enjoyment or exercise on an equal basis with others of all human rights and fundamental freedoms;

Making it proportionate :)

Regression testing

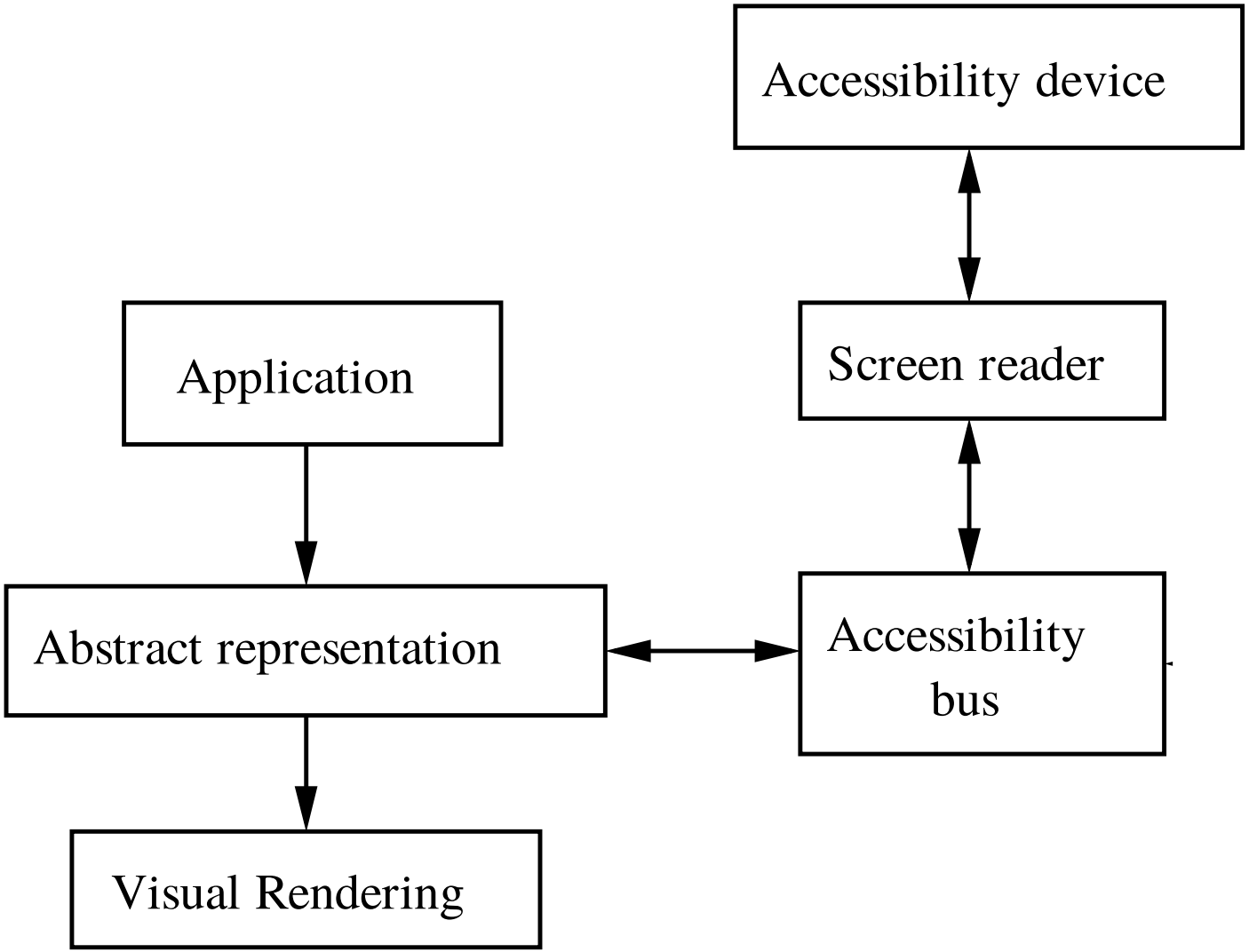
- For a start, avoid new issues
  - Automatically point them out
  - Teach programmers about them exactly when writing the code
- While still showing what needs to be done
  - “Existing issues” list, to be worked on too
    - But as a second priority

What are we talking about, technically?

Design principles

- Same software, made accessible
  - Understand each other, get help, etc.
  - Notably for big software
    - Can’t reimplement accessible versions
    - Leading FOSS
- Synchronized work
  - Just alternate input/output
  - Being able to work together
- Pervasive
  - Shouldn't have to ask for software installation / configuration

Generic methodology



Abstract representation

- Window
  - Vertical container
    - Menu bar
      - File Menu
        - Open Menu Item
        - …
      - ...
    - Horizontal container
      - Text area
      - Ok button

Technically speaking

A lot of applications already *technically* accessible

- Console
- GTK2/3
- KDE-Qt4 sketchy, Qt5 improving
- Java Swing
- Acrobat Reader

A lot are not

- Mono?
- KDE-Qt3
- Xt
- Self-drawn (e.g. xpdf)

In practice

- Technically-accessible doesn't mean usable
  - A visually-organized mess of widgets...

First name:	Foo
Last name:	Bar
Password:	baz

In practice

- Technically-accessible doesn't mean usable
  - A visually-organized mess of widgets...

First column
- Label1 First Name
- Label2 Last Name
- Label3 Password

Second column
- Text1 Foo
- Text2 Bar
- Text3 baz

In practice

- Technically-accessible doesn't mean usable
  - A visually-organized mess of widgets...

- Label1 First Name for Text1 Foo
- Label2 Last Name for Text2 Bar
- Label3 Password for Text3 baz

In practice

- Technically-accessible doesn't mean usable
  - A visually-organized mess of widgets...

First column
- Label1 First Name
- Label2 Last Name
- Label3 Password

Second column
- Text1 Foo
- Text2 Bar
- Text3 baz

In practice

- Technically-accessible doesn't mean usable
  - A visually-organized mess of widgets…

First column
- Label1 First Name
- Label2 Last Name
- Label3 Password

Second column
- Text1 Foo
- Text2 Bar
- Text3 baz

Screen reader "Script" for each application?

✗ Not viable, screen reader becomes ugly

✗ Unmaintainable with each new release

In practice

- Technically-accessible doesn't mean usable
  - A visually-organized mess of widgets…

First column
- Label1 First Name
- Label2 Last Name
- Label3 Password

Second column
- Text1 Foo labeled by Label1
- Text2 Bar labeled by Label2
- Text3 baz labeled by Label3

➔ Rather declaration of relationships between widgets

Graphical applications

- Design your application **without** gui in mind first
  - Logical order, just like CSS ☺
- Use standard widgets
  - e.g. *labeled* text fields
  - Avoid homemade widgets, or else implement atk
  - Provide alternative textual content for visual content
- Make sure of logical relations between widgets
- Keep it simple!
  - screen reading easier, but also for everybody!

Test it yourself! (GUIs)

orca -e braille-monitor

- Then work as usual
- Only using keyboard
- Checking proper text appears there

And crash-test

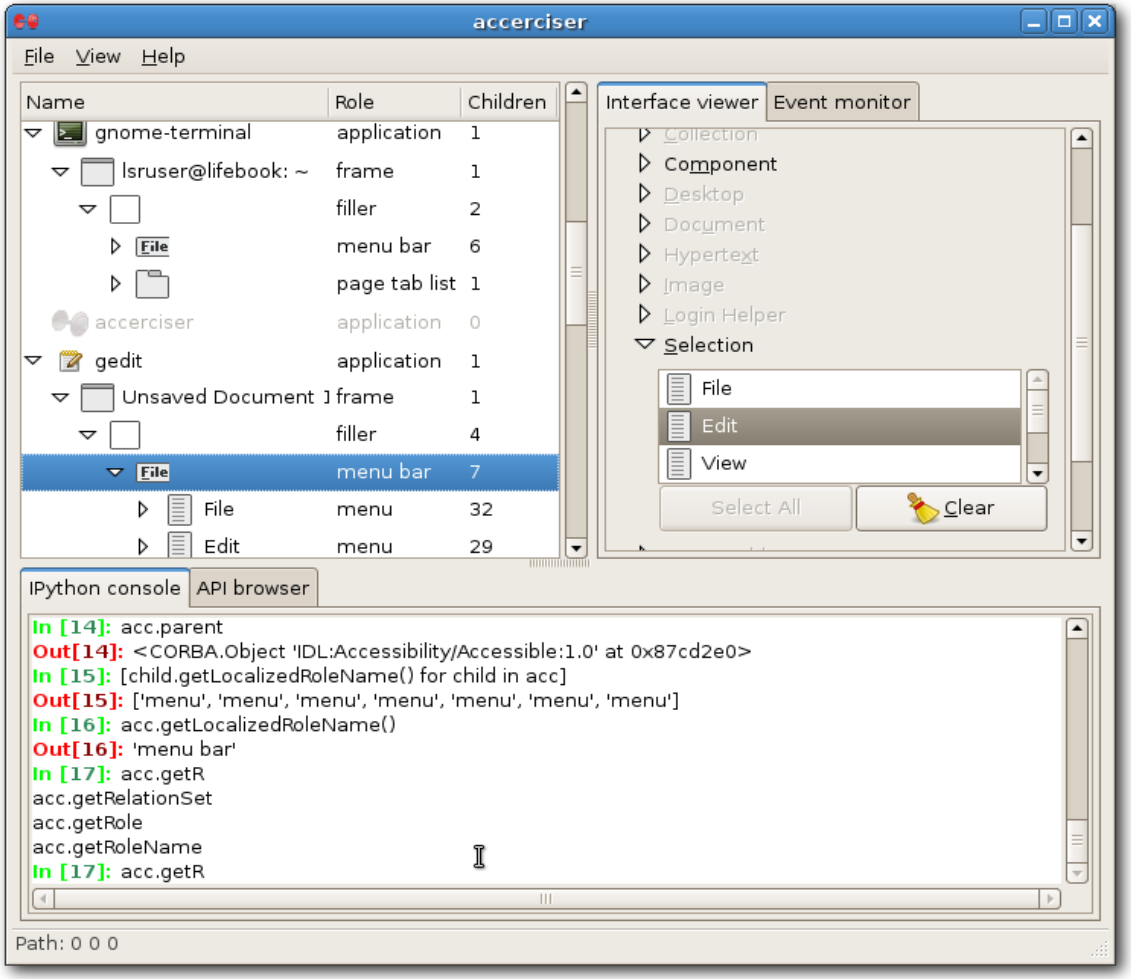
- Turn on speech, switch off the screen

https://developer.gnome.org/accessibility-devel-guide/stable/

Test it yourself! (GUIs)

Accerciser

- Sort of debugger
- Tree of widgets
- Properties



https://developer.gnome.org/accessibility-devel-guide/stable/

Test it yourself? Really?

Well, maybe that's not proportionate…

- Teach every programmer to use Orca & accerciser?
- Programmers always using them when developping?

Rather automatic tests in CI

- Avoid new issues
- Have a FIXME list
- Still make people aware of the question on everyday development like security & i18n

Regressions

- One step forward, two steps backward
- New features break old features
- Bug fixes break features
- So many potential regressions

- Regression testing
- User testing

Accessibility regressions

- One step forward, ten steps backward
- New features just unusable
- Bug fixes break usability

- User testing
  - So few concerned users
  - Difficult to test development versions
  - Developers have no idea how to test it

➔ Regression testing?

Regression testing

New features: making sure they might be usable

Old features: making sure they remain usable

- All widget have some label
  - Except layout widgets & such
- All labels are attached to some widget
  - Probably there visually for a reason...
- All widgets are reachable with some keyboard shortcut

Testing widget labelling

Online, observing running application from outside?

- Expensive
- Relating results with source line?

Runtime-tests within toolkit?

- Raise warnings
- Relating results with source line?

R&D

Testing widget labelling

Static analysis with C/C++/… source code

- Uh
- Source parsing?
  - Uh

R&D

Testing widget labelling

With glade UI (XML files)

- “Lint” tool which browses XML widget trees

```
$ gla11y $(find . -name \*.ui)

./sw/uiconfig/swriter/ui/sortdialog.ui:140 WARNING:
'GtkLabel' 'order' has no label-for nor mnemonic_widget

./sw/uiconfig/swriter/ui/sortdialog.ui:156 WARNING:
'GtkRadioButton' 'up1' is missing a label

./sw/uiconfig/swriter/ui/tocindexpage.ui:1012 ERROR: 'svxcorelo-
SvxLanguageBox' has multiple mnemonic_widget: lines 588, 612, 1004

./sw/uiconfig/swriter/ui/mmsalutationpage.ui:314 ERROR:
mnemonic_widget 'fieldname' not found
```

Easy to embed in Makefiles

Testing widget labelling

On LO 5.2.7, current rough result: ~8000 warnings

- ✔ Use “suppression” files
  - Starting regression database with **0** warning
- ✗ False positives
  - Automate some classes of suppressions
  - Mark others

Will be working on that this month (Feb 2018)

- By Hypra & Martin Pieuchot with TDF funding

https://git.hypra.fr/youpi/libreoffice-non-regressions

Conclusion

- Accessibility is a concern for a lot of people
  - 10% have major concerns
  - 20% have minor concerns
- It very often actually also helps other users
- But we need to raise awareness of this
  - Regression testing can be a vector for that
  - Integrate a11y in the development processes
    - To actually enforce it
    - Like security & internationalization
- Will also work on keyboard shortcuts later